Outline

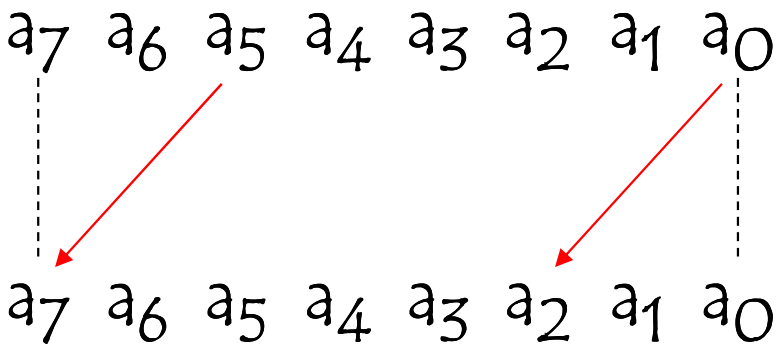
- Digital CMOS design
- Arithmetic operators
  - Adders
  - Comparators
  - Shifters
  - Multipliers

Shifters

Shifting a value

Let consider a value a coded on 8 bits

a can be shifted to the left by n positions ($0 \leq n < 8$)



For a natural number, shift left is a multiplication by 2^n

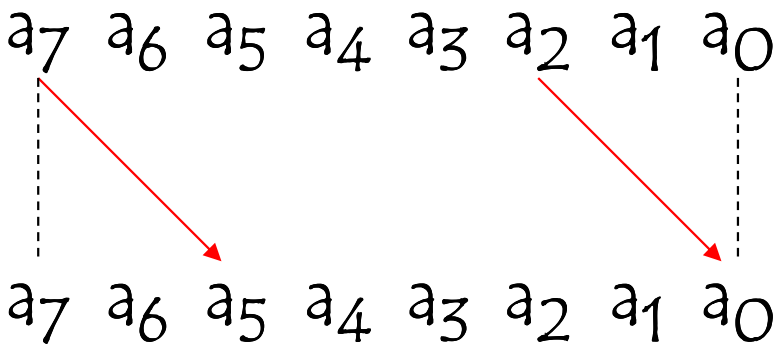
Shifters

Shifting a value

Let consider a value a coded on 8 bits

a can be shifted to the right by n positions ($0 \leq n < 8$)

logic



For a natural number, shift right is a division by 2^n

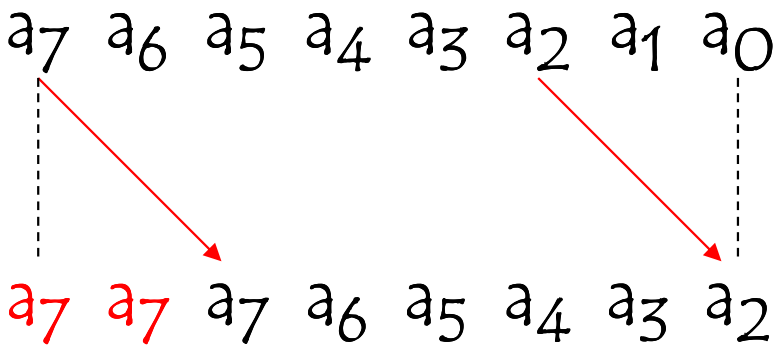
Shifters

Shifting a value

Let consider a value a coded on 8 bits

a can be shifted to the right by n positions ($0 \leq n < 8$)

arithmetic



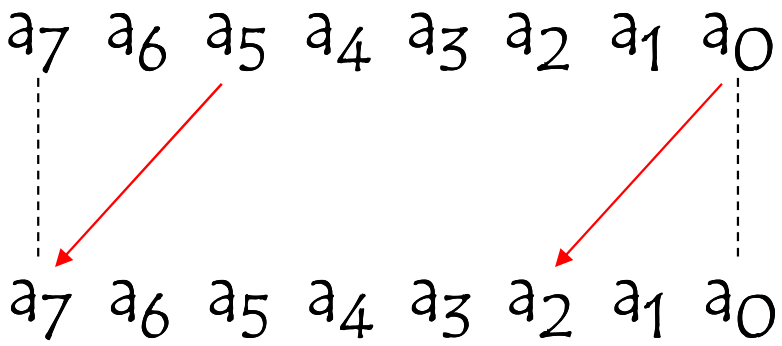
For a relative number, shift right is a division by 2^n

Shifters

Shifting a value

Let consider a value a coded on 8 bits

a can be rotated to the left by n positions ($0 \leq n < 8$)

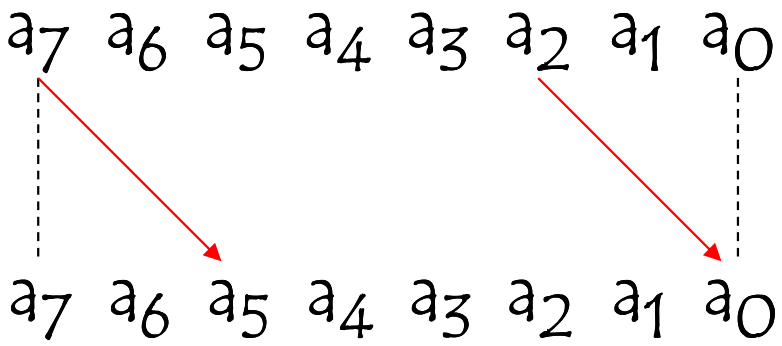


Shifters

Shifting a value

Let consider a value a coded on 8 bits

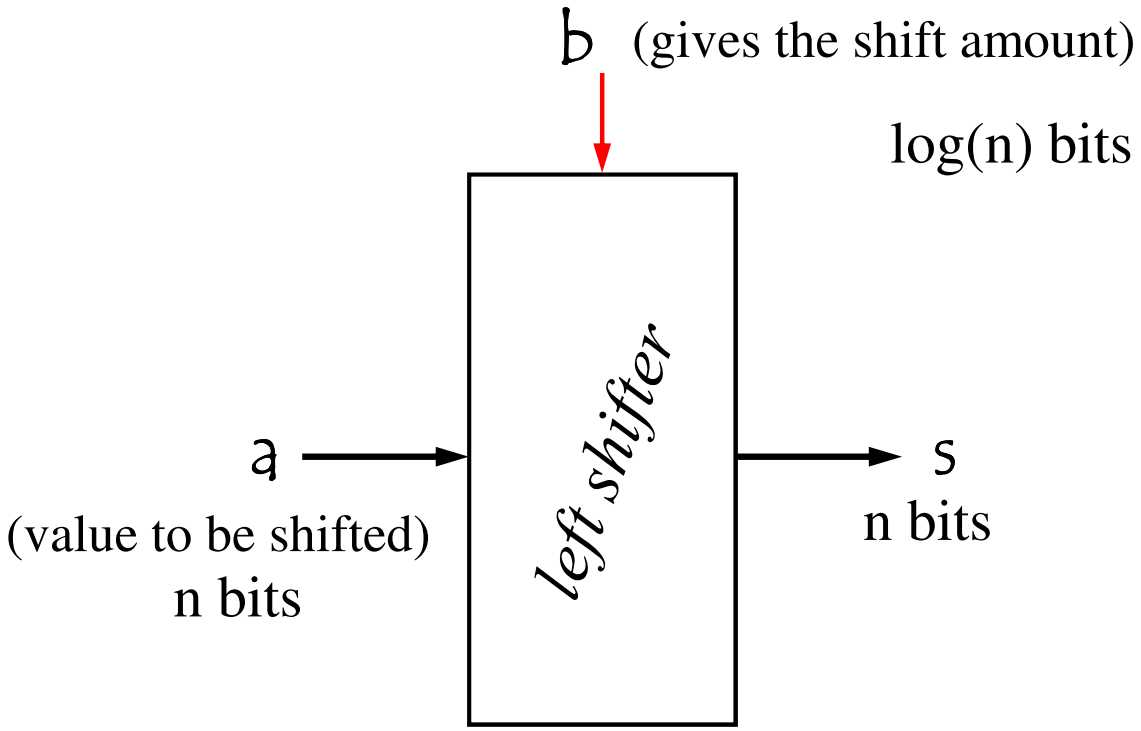
a can be rotated to the right by n positions ($0 \leq n < 8$)



Shifters

Shifting a value

Implementation (left shifter)



Shifters

Shifting a value by 1 position to the left

Boolean function

b coded on 1 bit

If $b = 0$

$s_i = a_i$

else

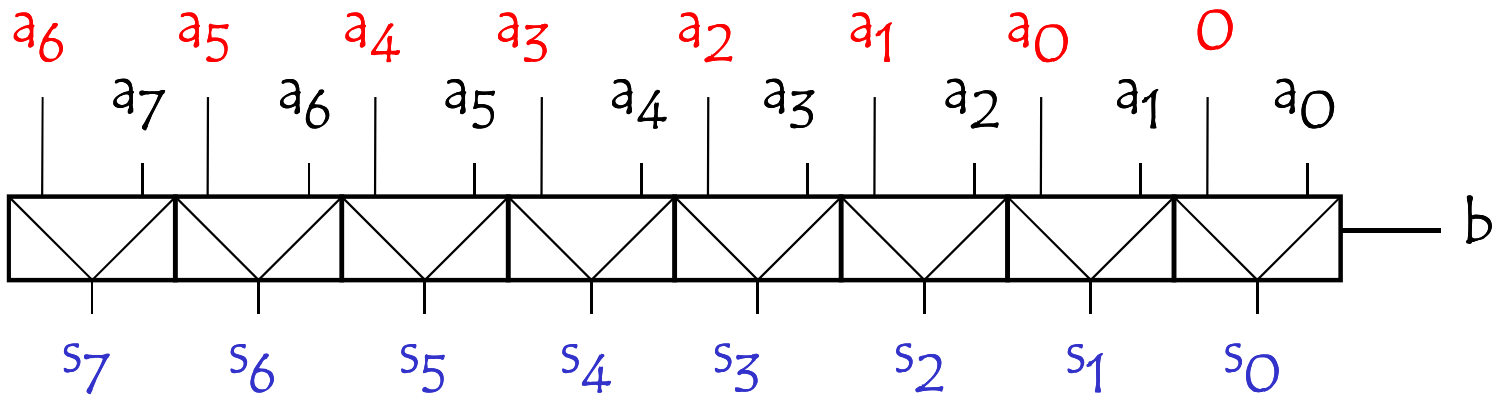
$s_i = a_{i-1}$ assuming $a_{-1} = 0$

$s_i = b.a_{i-1} + \overline{b}.a_i$ ⇨ 2-input Multiplexer

Shifters

Shifting a value by 1 position to the left

Implementation



Shifters

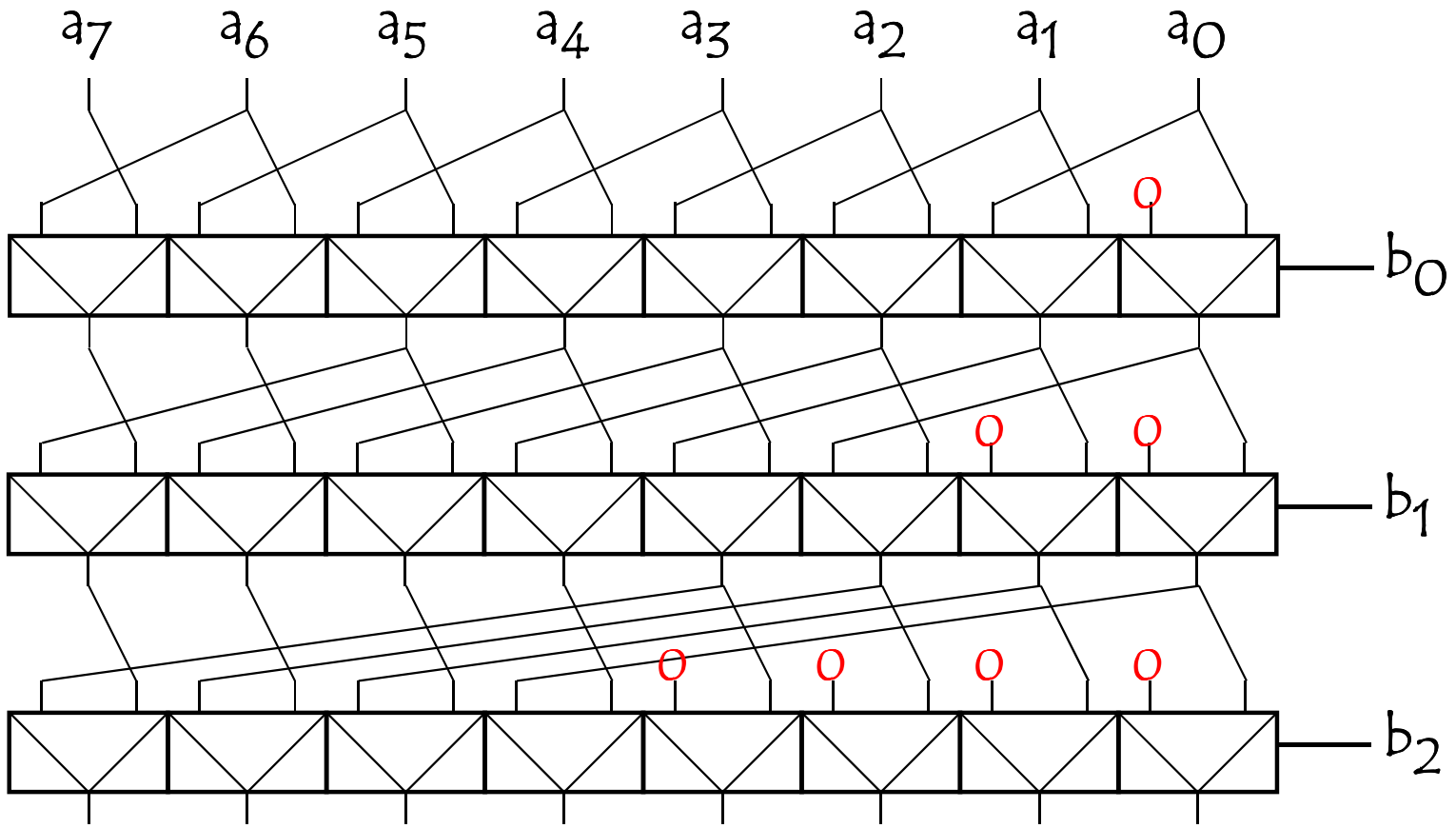
Shifting a value by b positions to the left

b may be 0 = *000*
1 = *001*
2 = *010*
3 = *011*
4 = *100*
5 = *101*
6 = *110*
7 = *111*

A combination of 2^2, 2^1, 2^0

Shifters

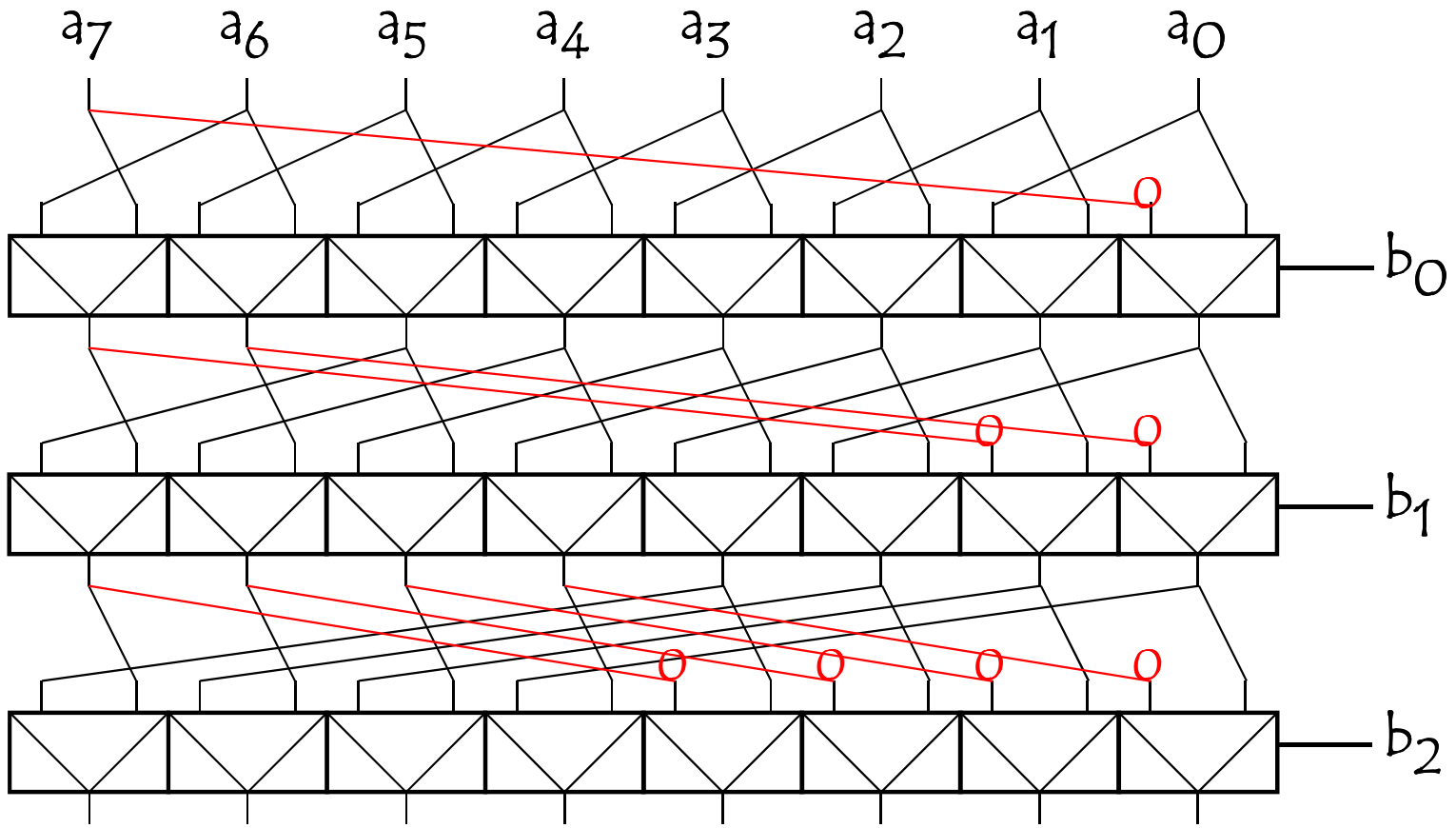
Shifting a value to the left



Area ∝ n *log*(n) Delay ∝ *log*(n)

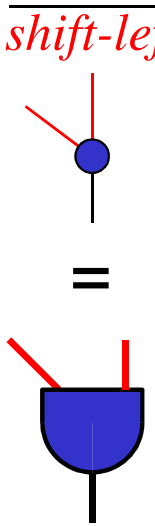
Shifters

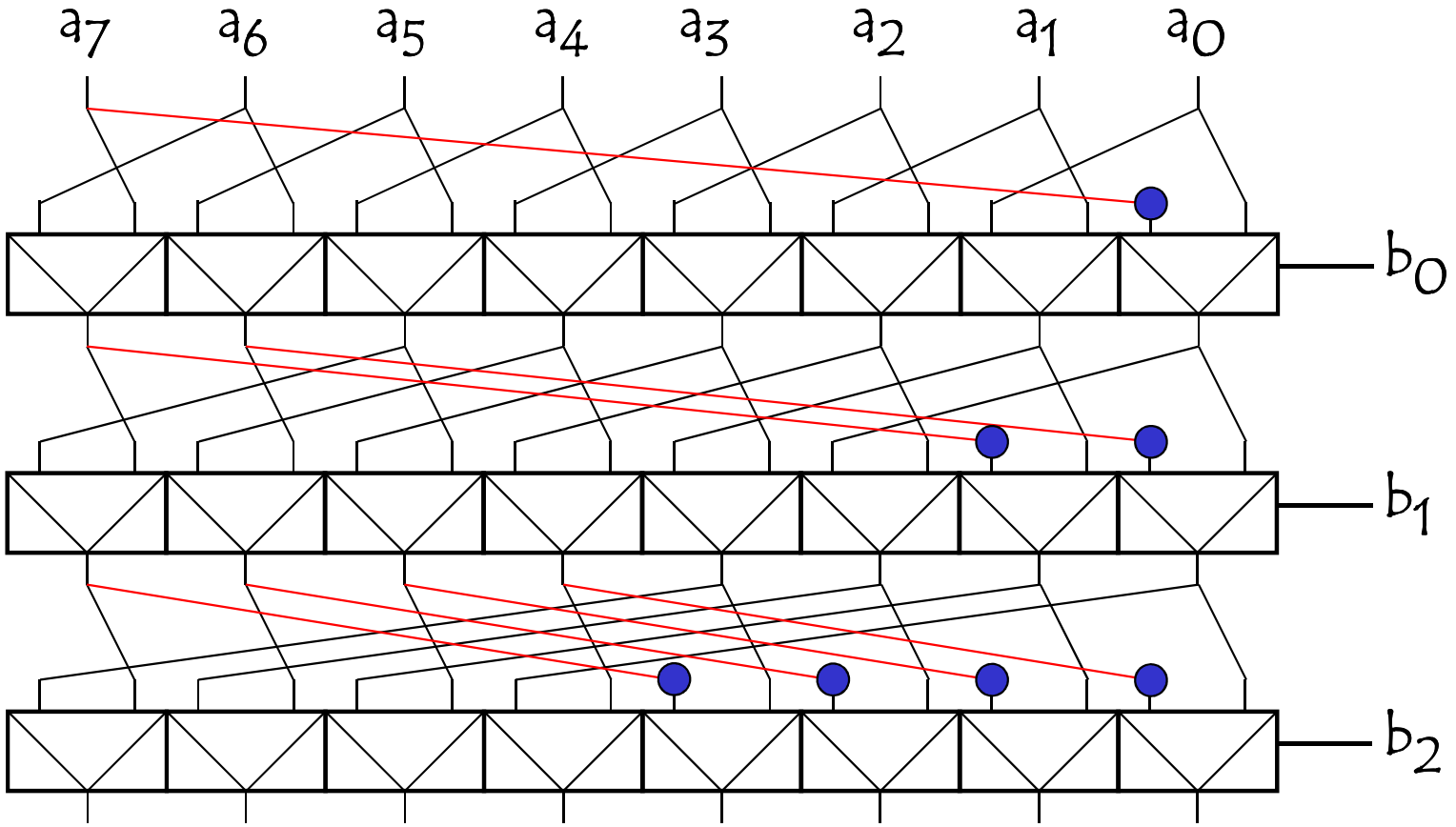
Rotate a value to the left



Shifters

Shift / Rotate a value to the left





Shifters

Shifting a value

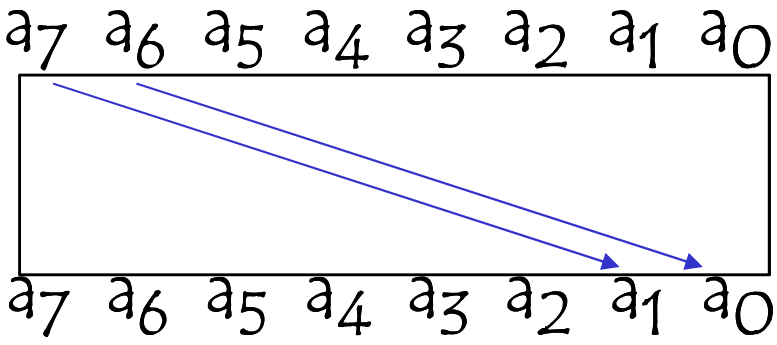
Rotate left by 2

=

Rotate right by 6

$6 = -2$

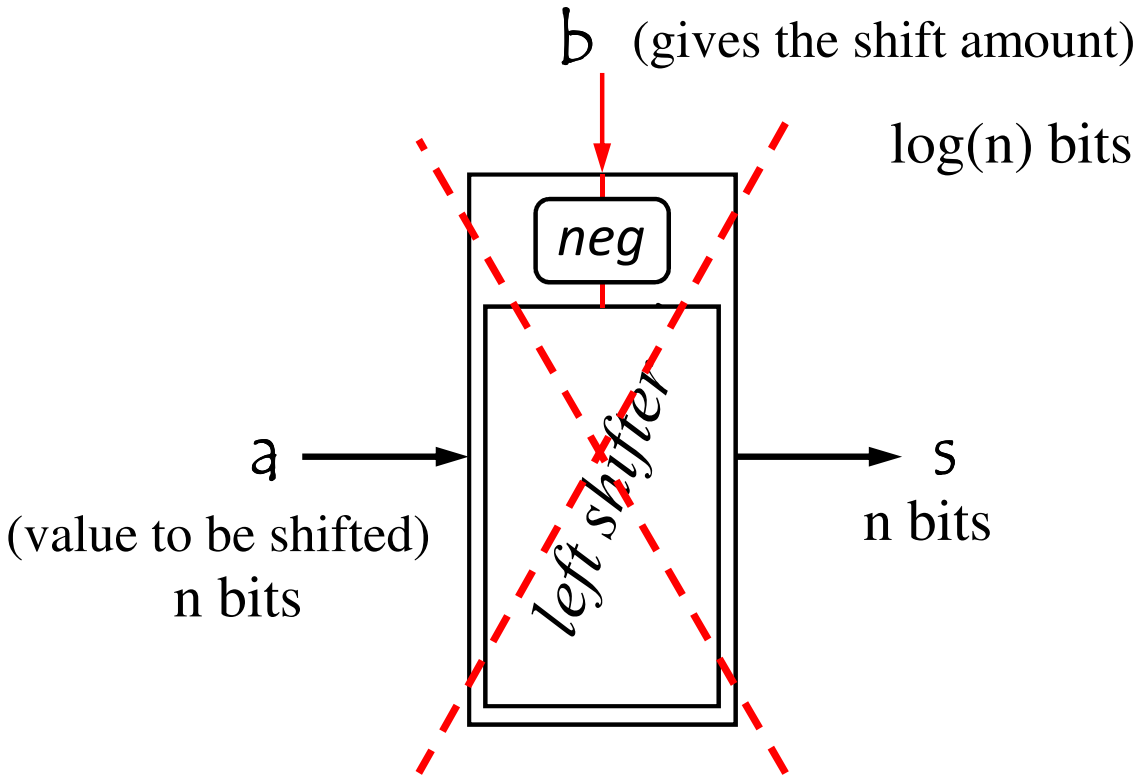
$110 = \overline{010} + 1$



Shifters

Shifting a value

Implementation (right shifter)



Shifters

Shifting a value

Rotate right by b

=

Rotate left by $-b$

=

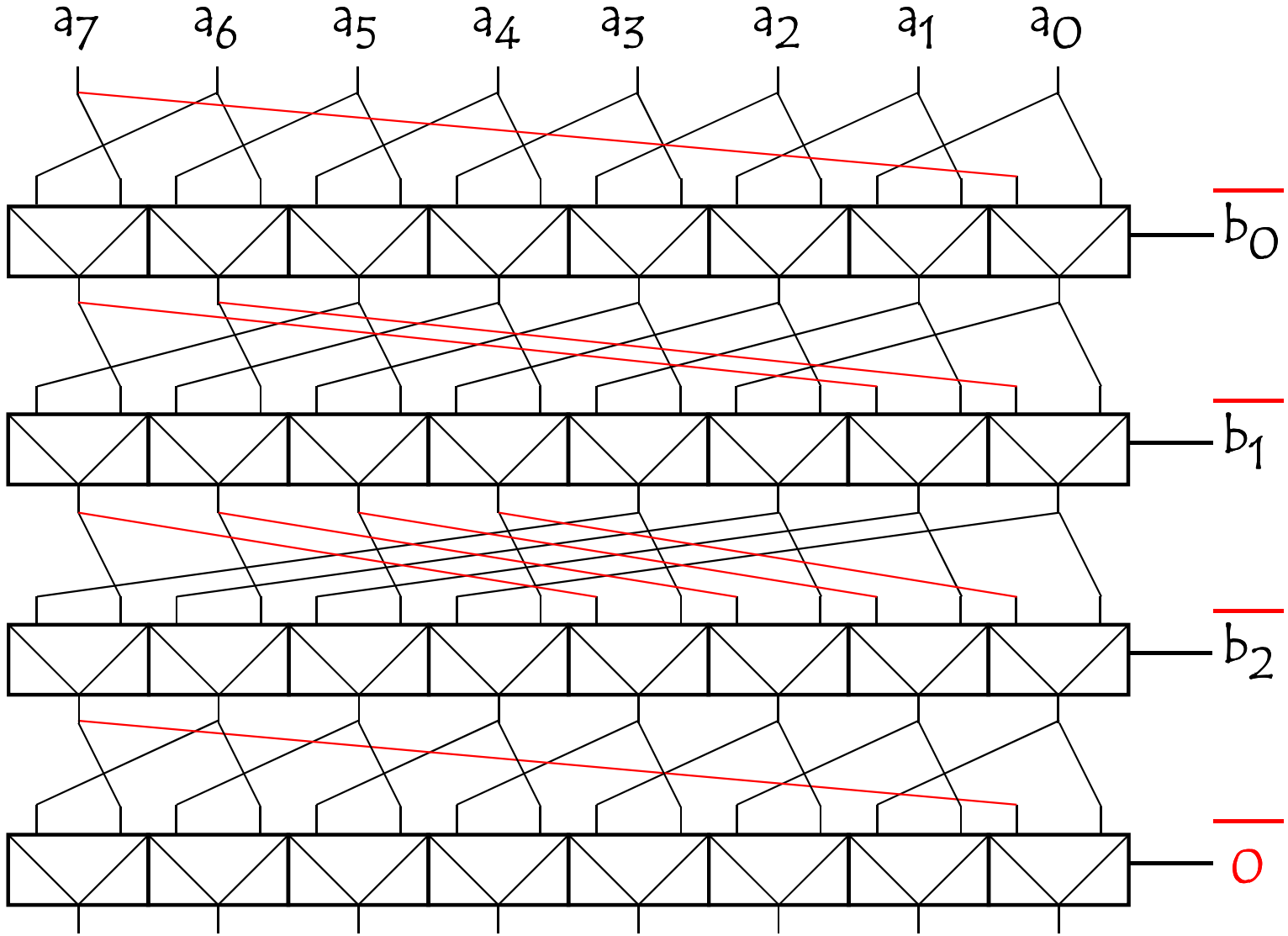
Rotate left by $(\overline{b} + 1)$

=

Rotate left by $\overline{b}$ followed by a 1 position rotate

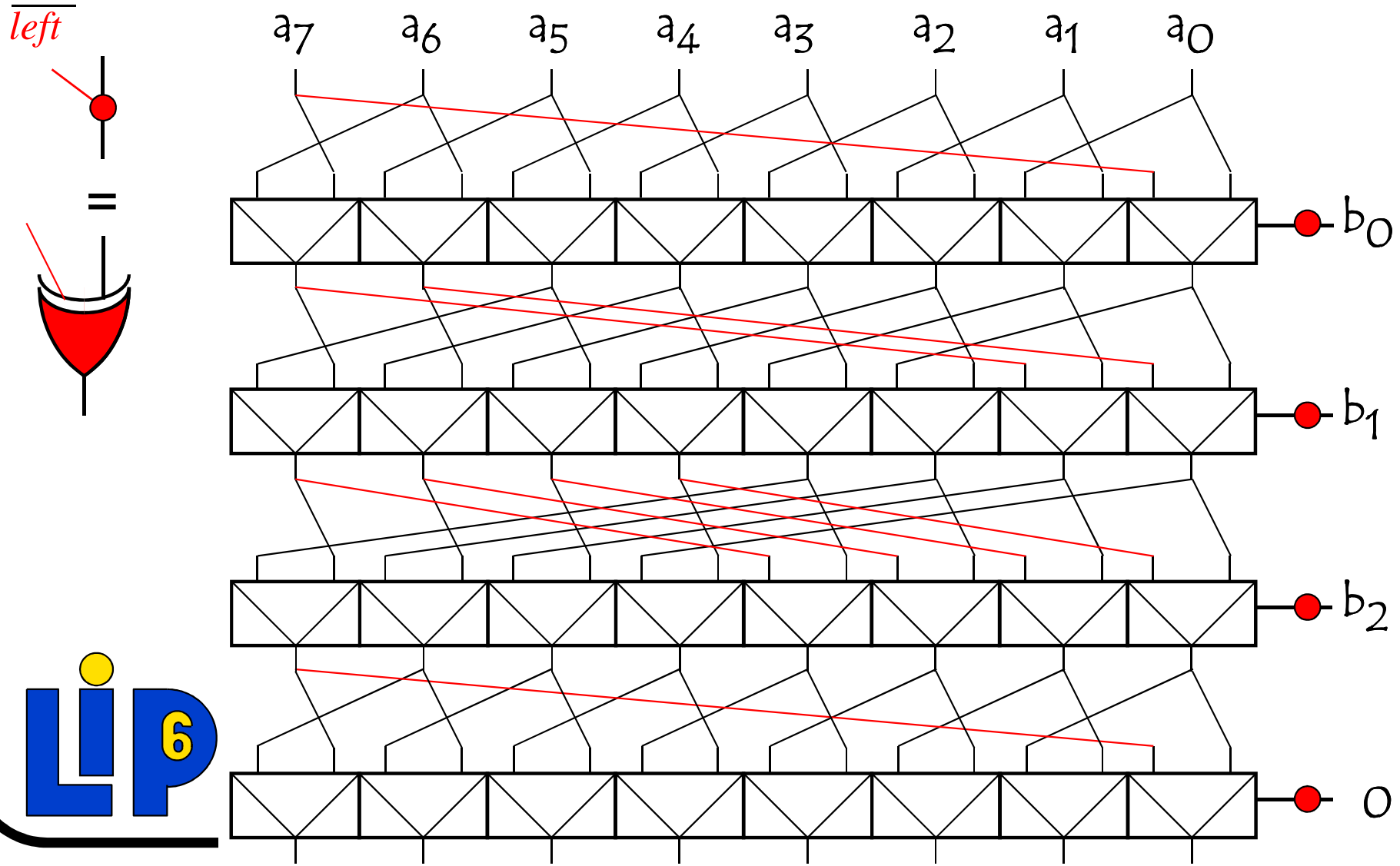
Shifters

Rotate a value to the right



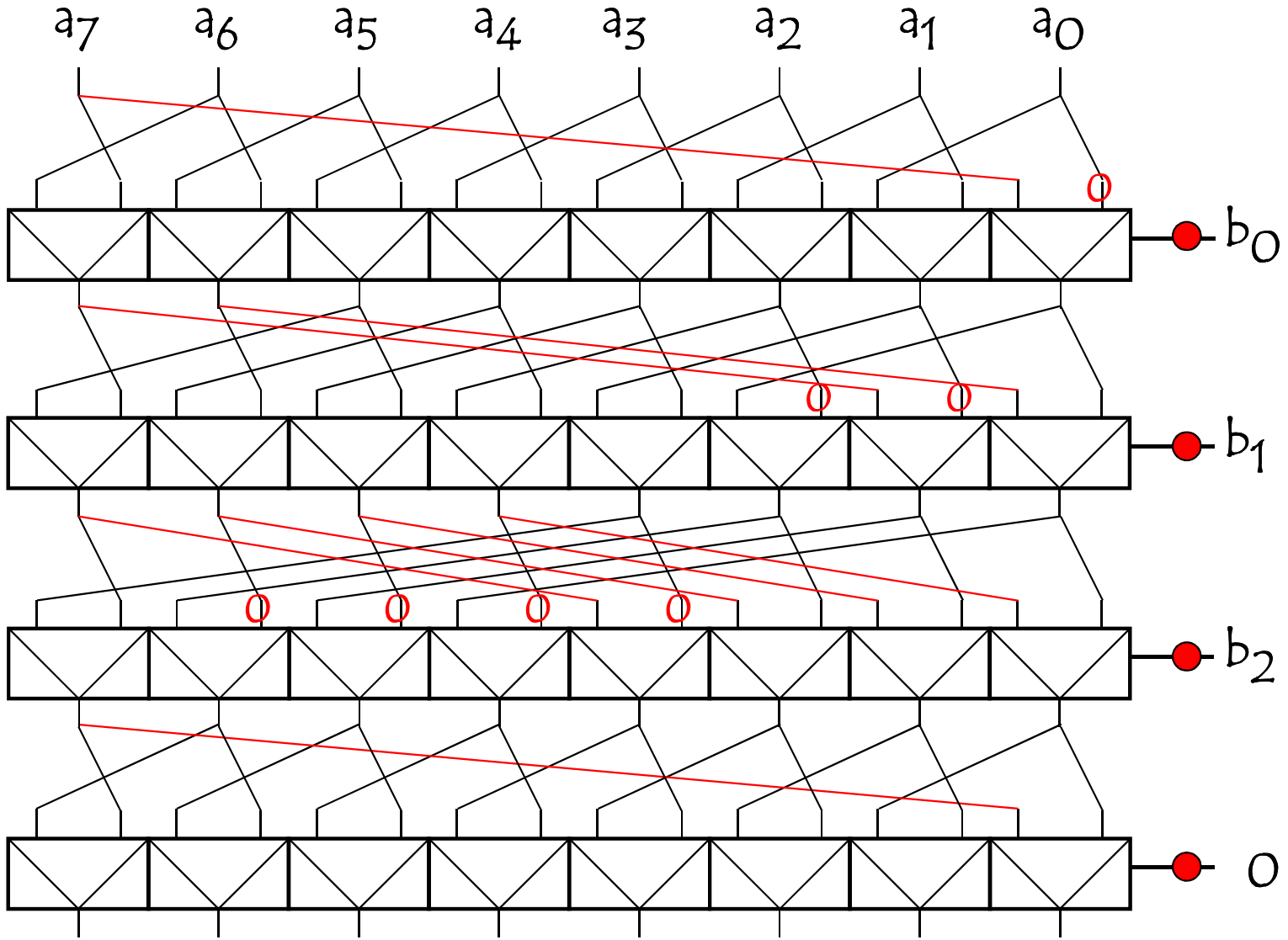
Shifters

Rotate a value to the left / right



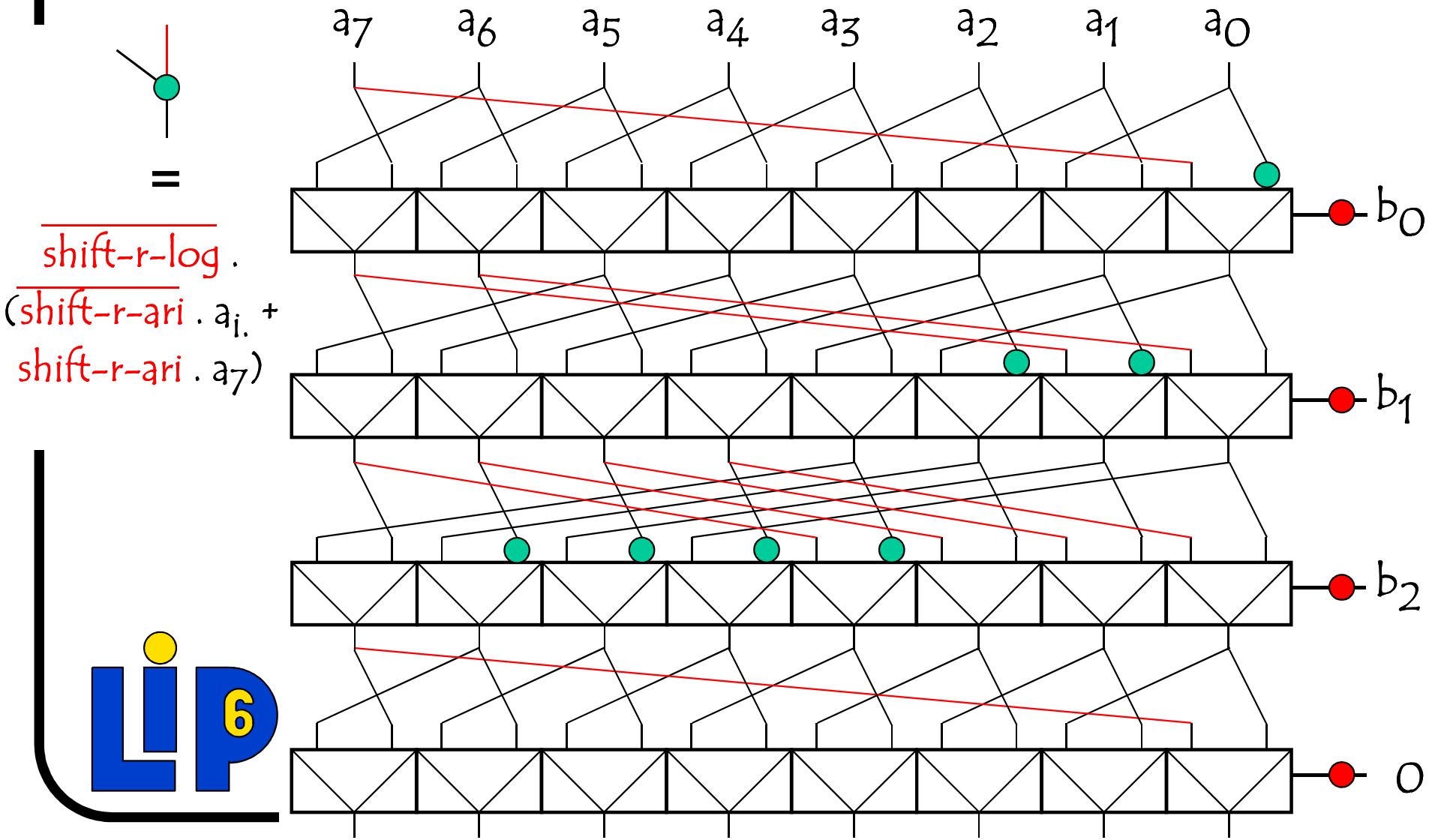
Shifters

Shift logic a value to the right



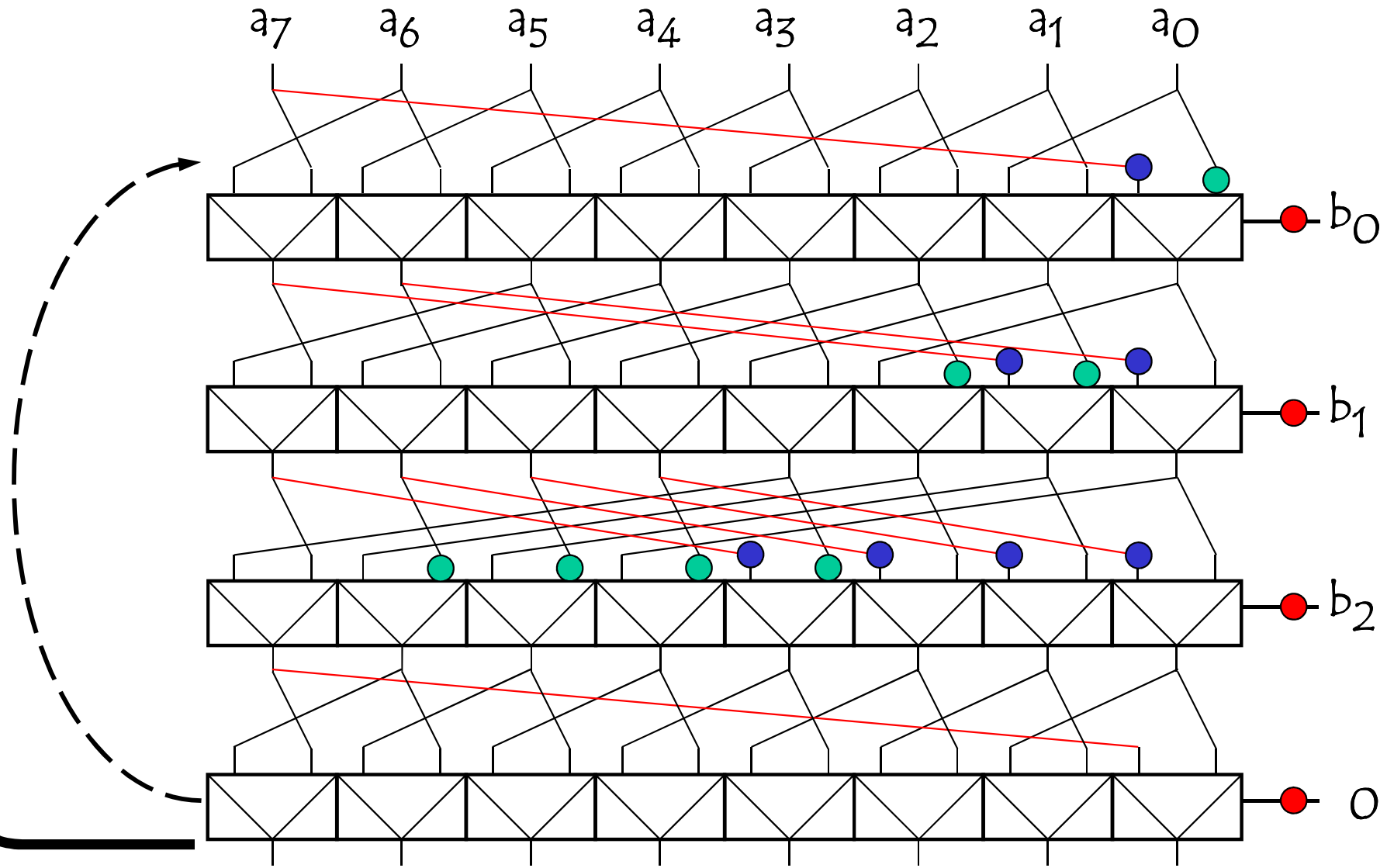
Shifters

Shift / Rotate a value to the right



Shifters

Shift / Rotate a value to the left / right



Shifters

Shift / Rotate a value to the left / right *optimized*